Using Active Directory to Authenticate Linux Users

Using Standard Protocols and Open Source Products

5/22/2006

Executive Summary

Goals and Objectives

In an ever changing world, technology is always changing with new technologies emerging everyday. Diverse technologies, like Linux and Microsoft® Windows, need to be integrated into heterogeneous networks and provide seamless and transparent services and resources to the end-user.

When our testing began we were given several key objectives. These objectives were:

1. To use Microsoft® Active Directory for user authentication and authorization.
2. To not have any shadowed accounts local to the workstation.
3. Make the process transparent to the user.
4. The ability for a new user to logon to the console.

Our goal is to achieve our objectives using the following authentication & authorization protocols:

- Light-weight Directory Access Protocol (LDAP)
- Kerberos version 5

Infrastructure and Resources

To achieve our goals and objectives we will use several products to achieve our overall solution. The products are as follows:

- Microsoft Windows Server 2003 R2 (Release Candidate 1)
- Microsoft DNS Server
- Microsoft Enterprise Certificate Server
- Microsoft Identity Management for Unix
- Microsoft Windows Server 2003 Support Tools
- Microsoft Windows Server 2003 Resource Kit Tools
- Open SUSE 10.0 Linux
    - Openldap
    - Name Server Switch Service (NSS)
    - Pluggable Authentication Modules (PAM)
    - OpenSSL
    - Kerberos v5 (MIT Implementation)
    - Samba v3

Lab Documentation

Active Directory Setup

For testing an Active Directory implementation was built on a VMware ESX server. This was done to take snapshots of the environment, so that a quick rollback could be done.

The forest was setup with an empty root as to duplicate the current production environment. A child domain will be created to hold user and computer accounts. The forest functional level is Windows 2003, and the domain functional level is Windows 2003 same as production. Each domain controller will be running DNS Server service and will contain and Active Directory-Integrated Zone for their domain.

Each server will be installed using Windows 2003 Server with SP1, which is required in order to install Windows 2003 Server R2. When the operating system is installed, a modification to the schema will need to be done before installing Windows 2003 Server R2. The Windows Server 2003 Support Tools and Resource Kit will also need to be installed for various tools that will be needed throughout our testing.

A Microsoft Enterprise Certificate Server will also need to be installed and configured to allow computer accounts to auto enroll for certificates.

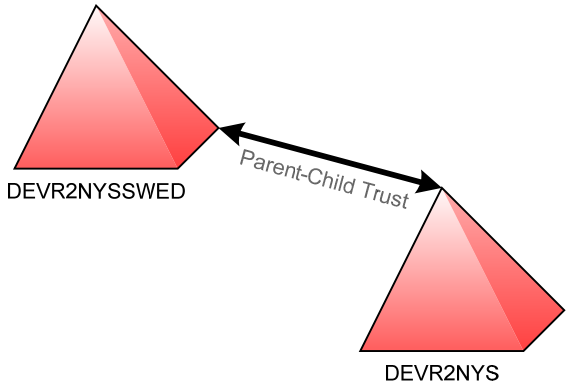

Figure 1 - Active Directory Diagram

Linux Configuration

For the Linux workstation a Dell Workstation installed with OpenSuSE 10.0 was used. When installing SUSE, the default workstation selection setup was selected; and the installer was allowed to auto partition the hard drive. A number of different PAM and NSS modules will need to be install for each method used.

Pluggable Authentication Modules (PAM)

Pluggable Authentication Modules, or PAM for short, is a mechanism used by software for authentication purposes. PAM provides a way to develop programs that are independent of authentication scheme, with a modular design due to its use of different system libraries. PAM is either configured by editing a single file named /etc/pam.conf or by editing individual files located in a directory named /etc/pam.d; depending on which version of Linux we are using. For each section of this document a list of which modules that were installed and how they were configured to work will be provided.

Name Service Switch Module (NSS)

The Name Service Switch module allows for the replacement of many UNIX configuration files with a centralized database. Among the files that it replaces are /etc/passwd, /etc/group and /etc/hosts. It was originally created by Sun Microsystems for Solaris, but has been ported to support other versions of UNIX and Linux.

NSS is usually configured by editing the /etc/nsswitch.conf file. For each service listed in this file, a list of one or more databases from which it can retrieve data.

LDAP

What is LDAP?

Lightweight Directory Access Protocol (LDAP) refers to a directory service that store objects with attributes in a hierarchical structure. Microsoft® Active Directory is used as the primary LDAP store by many organizations. Active Directory is accomplished with one or more domain controllers, each with its own writable copy of the LDAP database. If more than one domain controller exists for a domain, then multi-master replication occurs to synchronize the databases on the separate domain controllers. Windows Server 2003 R2 has a modified schema that allows for the mapping of UNIX attributes to new Active Directory attributes. Without this updated schema this method would not be possible.

Required Linux Modules

The Linux workstation will need to have installed the appropriate libraries and modules. The following modules will be required for LDAP authentication:

- openldap-client-2.2.27-6
- ldapcipplib-0.0.3-33
- nss_ldap-238-2
- pam_ldap-178-3
- libnscd-1.1-5
- nscd-2.3.5-40

These modules can be installed by using the Software Management application located within the open SUSE Yast Control Center.

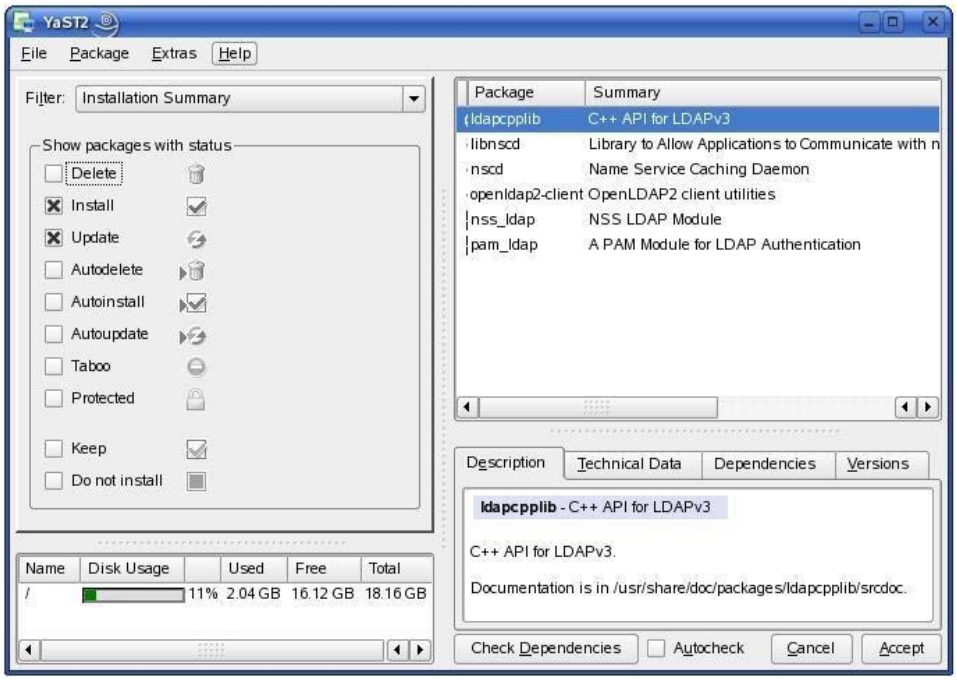

Figure 2 - LDAP Modules

Configuring /etc/openldap/ldap.conf

In SUSE Linux there are two ldap.conf files. The /etc/openldap/ldap.conf file is used by the client libraries and tools. The other is /etc/ldap.conf which is used for the PAM and NSS modules.

In the /etc/openldap/ldap.conf file, we will need to edit both the HOST and BASE lines of the file. The host should designate the FQDN of a domain controller for the domain. The BASE attribute will designate the beginning of the search path within LDAP. It should look like the following:

```
HOST devr2dcaa.devr2.local
BASE cn=Users,dc=devr2,dc=local
```

Now to test the configuration, this is done using the ldapsearch command. The following command will test basic connectivity to Active Directory:

```
ldapsearch –x –s base –b “” “(objectclass=*)”
```

If successful, we should receive an output describing the Active Directory LDAP description. If not successful, check the IP and DNS configuration.

Active Directory Configuration

At this point, a successful anonymous connection to Active Directory has been made. Before proceeding any further a decision must be made whether to allow anonymous binds to Active Directory? By default, Active Directory will only allow anonymous binds to RootDSE; an anonymous search to the directory we will result in an error.

There are two choices, modify Active Directory to allow for anonymous connections, or pre-configure a restricted account for the workstations to use to search for users. Configuring a restricted user account is securest of the two options. If anonymous binds are chosen, refer to Microsoft knowledgebase article Q326690 - Anonymous LDAP operations to Active Directory are disabled on Windows Server 2003 domain controllers.

In this configuration, a user account named PADL that belongs to just “Domain Users” was created. This account will need to be given “Unix Attributes”, of an NIS domain, UID, Login Shell, and Primary Group; in *Active Directory Users and Computers*. Figure 3 shows an example of the settings used.

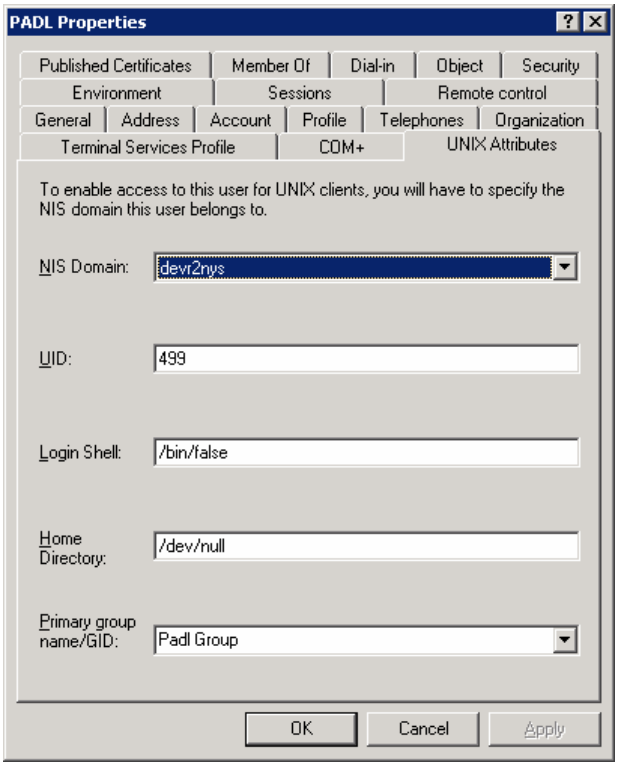

Figure 3 - PADL Account Settings

Configuring /etc/ldap.conf

Now that the account has been created and configured continue on by editing the /etc/ldap.conf. Remember, this file is used for the PAM and NSS modules.

```
host                    devr2dcaaaa.devr2.local
base                    cn=Users,dc=devr2,dc=local
ldap_version            3
binddn                  cn=PADL,cn=Users,dc=devr2,dc=local
bindpw                  MicrosoftLinux
ssl                     no
port                    389
scope                   sub
timelimit               30
nss_map_objectclass     posixAccount User
nss_map_objectclass     shadowAccount User
nss_map_attribute       uid sAMAccountName
nss_map_attribute       uidNumber uidNumber
nss_map_attribute       gidNumber gidNumber
nss_map_attribute       cn sAMAccountName
nss_map_attribute       uniqueMember msSFU30PosixMemberOf
nss_map_attribute       userPassword unixUserPassword
nss_map_attribute       homeDirectory unixHomeDirectory
nss_map_attribute       loginShell loginShell
nss_map_attribute       gecos name
nss_map_objectclass     posixGroup Group
pam_login_attribute     sAMAccountName
pam_filter              objectclass=User
pam_password            ad
nss_base_passwd         cn=Users,dc=devr2,dc=local?sub
nss_base_shadow         cn=Users,dc=devr2,dc=local?sub
nss_base_group          cn=Users,dc=devr2,dc=local?sub
```

The *host* entry defines the name of a domain controller, for a production environment, list a number of domain controllers separated by a space, or just enter the domain name of which a domain controller would be returned from DNS.

On the next line is *base*, which is used to define were in Active Directory to start looking for objects. This can be used to limit the ability for certain users contained within a certain set of OUs to login to a Linux workstation.

By default, PAM will assume a version 3 LDAP source, we define it with the next line *ldap_version* to clarify.

Binddn is defined so that PAM has an account to use in order to query Active Directory. As outlined earlier in this section, either use this method or modify Active Directory to allow anonymous binding to all of Active Directory. This account is given "domain user" and nothing else. With further configuration one could lockdown this account, by giving it read only access to the required objects. *Bindpw* is the password for the account used in binddn.

SSL would be used to encrypt traffic from the client to Active Directory. This will be covered later in this paper; at this point set this option to "no". *Port* if left blank will default to 389, this is just for confirmation.

The *scope* attribute confines the search to the base, one level below, or to search through all lower levels. This is done by entering the word base, one, or sub after the question mark.

Now comes a number of entries starting with nss. These entries map settings for the Name Service Switch Module. This maps essential Active Directory attributes to required UNIX attributes.

The last three references are for the PAM authentication. *Pam_login_attribute* defines what the user will use to logon with, since we have it defined to use the sAMAccountName; they can use their regular username for logon purposes. *Pam_filter* filters out user accounts to authenticate users against, not other Active Directory objects. Lastly, *pam_password* defines the password change mechanism to be used. This must be set to "ad", otherwise issues may arise when a user attempts to change their password from the Linux workstation.

Parameters highlighted in blue are dynamic; these will change to fit the environment the system in.

Configuring /etc/nsswitch.conf

In the nsswitch.conf add the keyword "ldap" to the following lines: passwd, shadow, group, hosts, networks, services, netgroup, and aliases. The file should look similar to the following when completed.

```
passwd:         files ldap
group:          files ldap
hosts:          files dns ldap
networks:       files dns ldap
services:       files ldap
protocols:      files
rpc:            files
ethers:         files
netmasks:       files
netgroup:       files ldap
publickey:      files
bootparams:     files
automount:      files
aliases:        files ldap
```

Once this file is edited and saved it can be tested to see if it works. The first test will be to see if it can see user accounts from Active Directory. To do this enter the command "getent passwd". It should enumerate all user accounts within the files, and then append all Active Directory accounts that have been configured with the correct UNIX attributes. It should look like the following example:

```
linux:~ # getent passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
daemon:x:2:2:Daemon:/sbin:/bin/bash
lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
mail:x:8:12:Mailer daemon:/var/spool/clientmqueue:/bin/false
news:x:9:13:News system:/etc/news:/bin/bash
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
games:x:12:100:Games account:/var/games:/bin/bash
man:x:13:62:Manual pages viewer:/var/cache/man:/bin/bash
at:x:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash
wwwrun:x:30:8:WWW daemon apache:/var/lib/wwwrun:/bin/false
ftp:x:40:49:FTP account:/srv/ftp:/bin/bash
postfix:x:51:51:Postfix Daemon:/var/spool/postfix:/bin/false
sshd:x:71:65:SSH daemon:/var/lib/sshd:/bin/false
ntp:x:74:65534:NTP daemon:/var/lib/ntp:/bin/false
mdnsd:x:78:65534:mDNSResponder runtime
user:/var/lib/mdnsd:/bin/false
messagebus:x:100:101:User for D-BUS:/var/run/dbus:/bin/false
haldaemon:x:101:102:User for haldaemon:/var/run/hal:/bin/false
nobody:x:65534:65533:nobody:/var/lib/nobody:/bin/bash
ldap:x:76:70:User for OpenLDAP:/var/lib/ldap:/bin/bash
PADL:x:499:499:PADL:/dev/null:/bin/false
_dsmith:x:10001:10000:Doug Smith:/home/_dsmith:/bin/sh
linux:x:10004:10000:linux:/home/linux:/bin/sh
linux:~ #
```

If this is not working review the previous steps for something that may be missing or mistyped.

Now that user accounts have been confirmed, lets try group mappings. For this use "getent groups". Again, it will enumerate all local groups stored in the local files and then append all the groups within Active Directory that are configured with UNIX attributes, as shown by the following:

```
linux:~ # getent group
root:x:0:
bin:x:1:daemon
daemon:x:2:
sys:x:3:
tty:x:5:
disk:x:6:
lp:x:7:
www:x:8:
kmem:x:9:
wheel:x:10:
mail:x:12:
news:x:13:
uucp:x:14:
shadow:x:15:
dialout:x:16:dsmith
audio:x:17:
floppy:x:19:
cdrom:x:20:
console:x:21:
utmp:x:22:
at:!:25:
public:x:32:
video:x:33:dsmith
games:x:40:
xok:x:41:
trusted:x:42:
```

```
modem:x:43:
ftp:x:49:
postfix:!:51:
maildrop:!:59:
man:x:62:
sshd:!:65:
ntadmin:!:71:
messagebus:!:101:
haldaemon:!:102:
nobody:x:65533:
nogroup:x:65534:nobody
users:x:100:
ldap:!:70:
Domain Users:x:10000:dsmith,_dsmith
Padl Group:x:499:PADL
linux:~ #
```

PAM Configuration

With successful authentication to Active Directory, authorization is the next step. This is done through editing of the PAM files. Before proceeding be warned to be careful, improperly configuring the PAM modules could leave cause you to be locked out of the system.

Below is the example of an open SUSE “xdm” PAM file. Add new authentication entries to check the local accounts, and then check ldap. For password it needs to be the same, which is used for changing passwords. Finally add a line in the session which tells PAM to create a local home directory on the workstation if one does not exist.

```
#%PAM-1.0
auth     sufficient pam_unix2.so nullok
auth     sufficient pam_ldap.so use_first_pass
account  include    common-account
password sufficient pam_unix2.so
password sufficient pam_ldap.so use_authtok
session  include    common-session
session  required   pam_mkhomedir.so skel=/etc/skel/umask=0022
session  required   pam_devperm.so
session  required   pam_resmgr.so
```

With all the changes made and saved, reboot. When the computer comes back up; test by using an Active Directory account to logon.

LDAP with SSL (LDAPS)

Why use SSL?

With normal LDAP authentication there is a major security concern, all information is transmitted between the client and server is in clear-text. In a secure environment this is not acceptable. Another issue that arises is that Active Directory will not allow a password change unless it can be done securely.

To solve these issues simply encrypt all LDAP requests using Secure Socket Layer (SSL). This will require an additional prerequisite of a Microsoft Enterprise Certificate Root Server.

A Microsoft Enterprise Certificate Root Server will need to be installed and configured to allow automatic computer enrollment for the domain. OpenSSL will need to be installed on the Linux workstation as well.

Installing an Enterprise Root Certificate Server

Perform the following steps to install and configure an enterprise CA on a Windows Server 2003 computer:

At a member server or domain controller in wer internal network, log on as a domain administrator. Click **Start**, point to **Control Panel** and click **Add/Remove Programs**.

In the Add or Remove Programs window, click the **Add/Remove Windows Components** button.

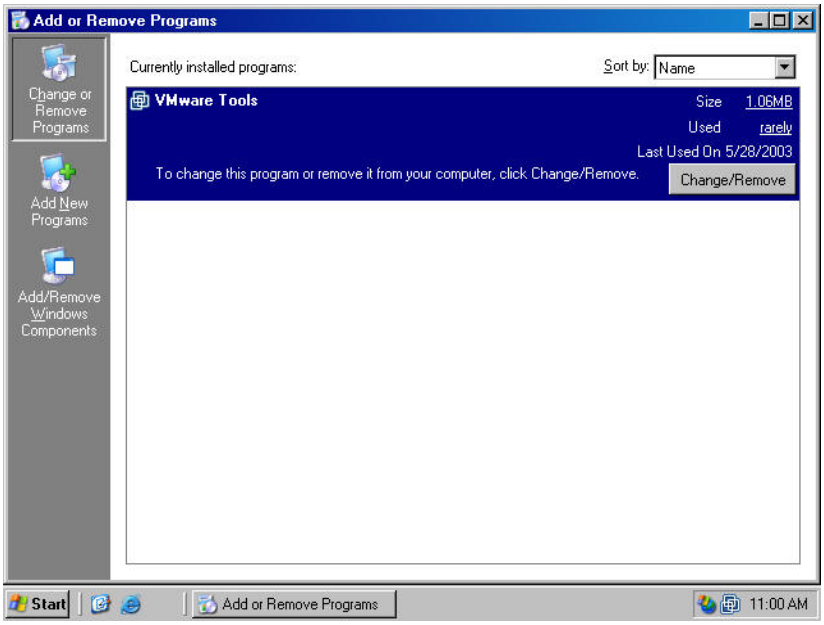

Figure 4 - Add/Remove Programs

In the Windows Components dialog box, click on the **Certificate Services** entry and click the **Details** button.

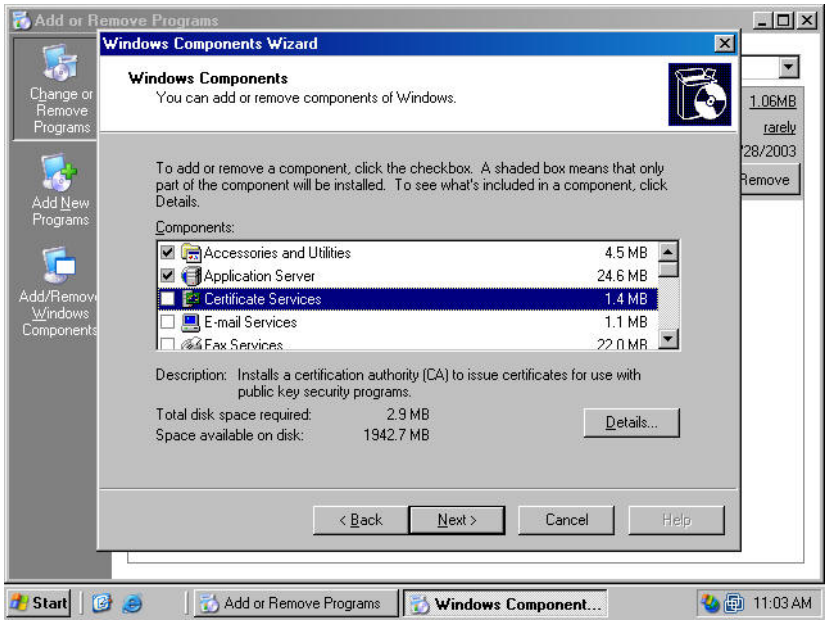

Figure 5 - Windows Components dialog box

In the Certificate Services dialog box, put a checkmark in the Certificate Services CA checkbox. A Microsoft Certificate Services dialog box appears and states that the machine name can not change as well as the domain membership of the machine while it acts as a certificate server. Read the information in the dialog box and click **Yes**.

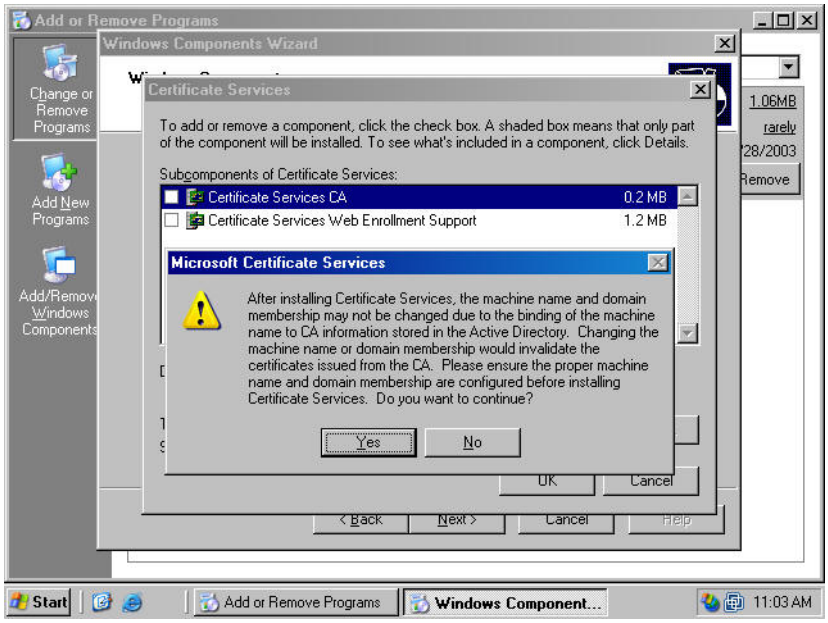

Figure 6 - Certificate Services dialog box

Both the Certificate Services CA and Certificate Services Web Enrollment Support checkboxes are checked. Click **OK** in the Certificate Services dialog box.

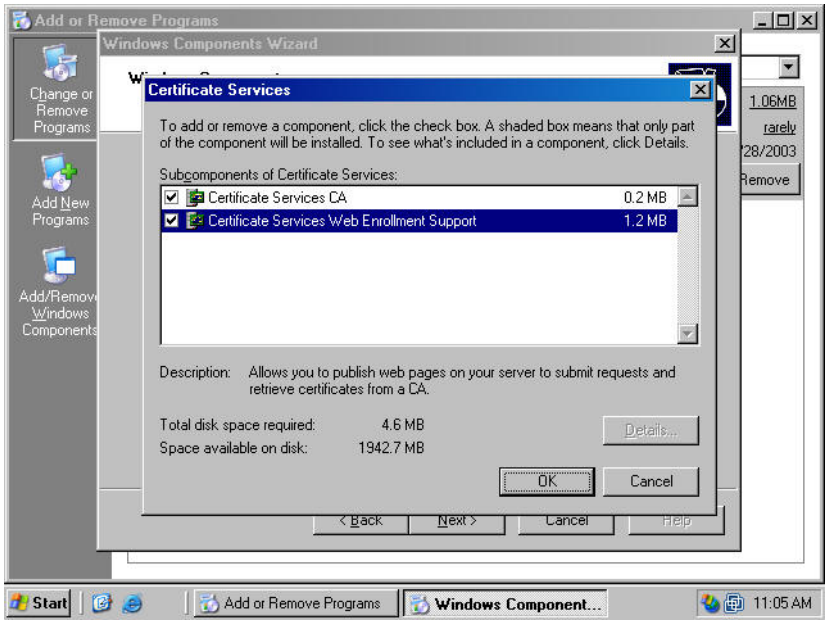

Figure 7 - Certificate Services dialog box

Click **Next** in the Windows Components dialog box.

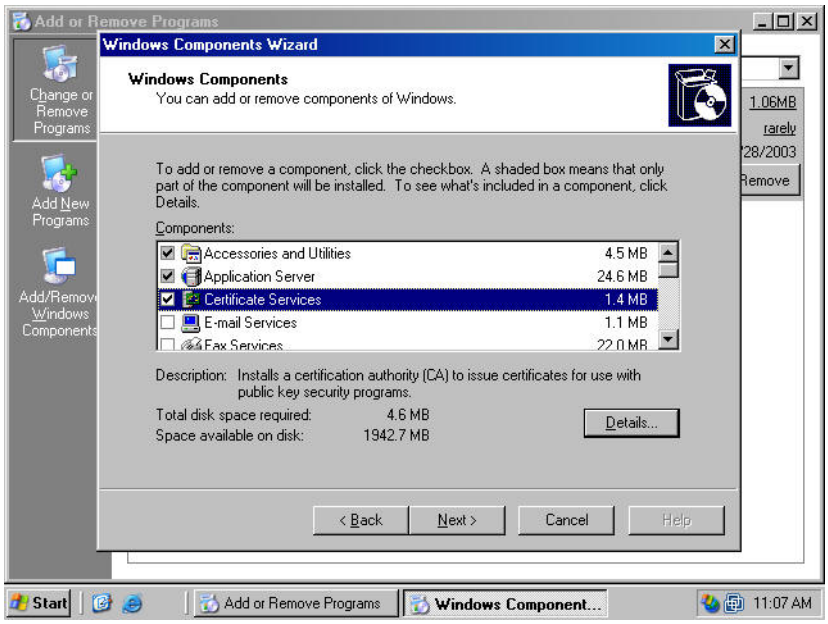

Figure 8 - Windows Components dialog box

Select the **Enterprise root CA** option on the CA Type page. Click **Next**.

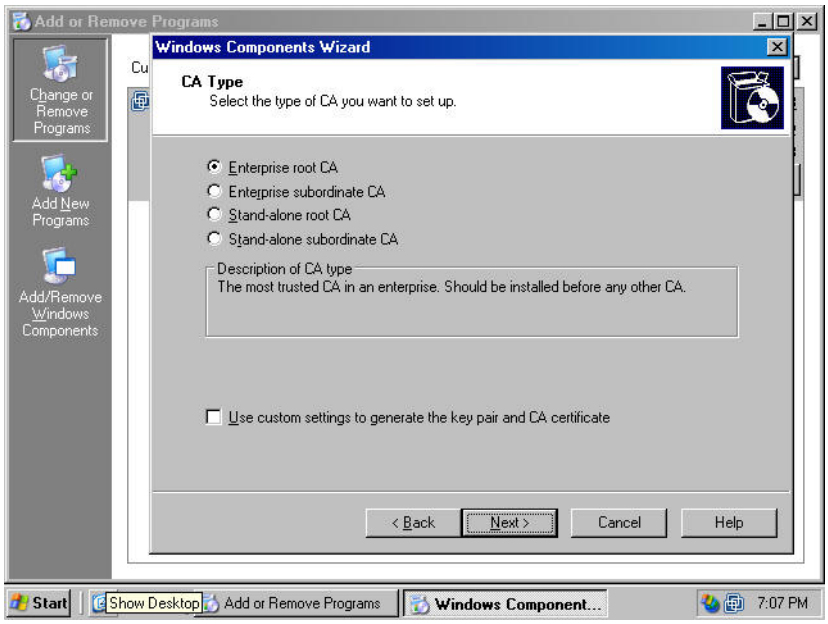

Figure 9 - CA Type page

On the CA Identifying Information page, type in a **Common name for this CA**. The common name of the CA is typically the DNS host name or NetBIOS name (computer name) of the machine running Certificate Services. The default **Validity Period** of the CA's self-signed certificate is 5 years. Accept this default value and click **Next**.

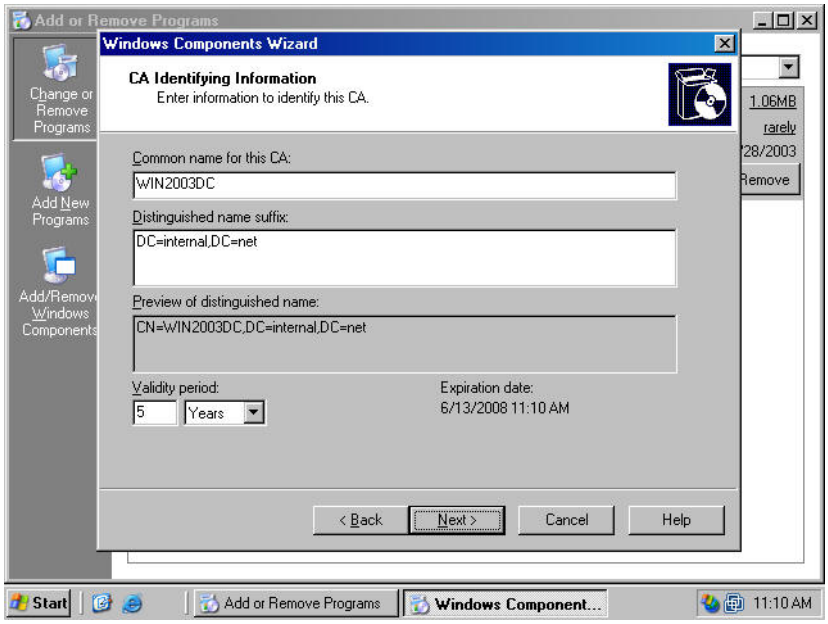

Figure 10 - CA Identifying Information page

On the Certificate Database Settings page, use the default locations for the **Certificate Database** and **Certificate Database Log**. Do not specify a shared folder to store configuration information because this information will be stored in the Active Directory. Click **Next**.

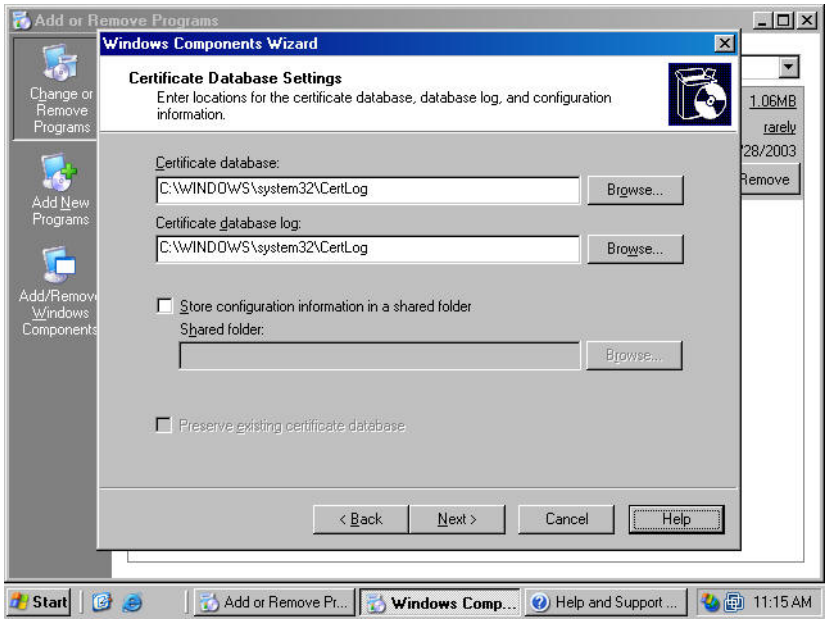

Figure 11 - Certificate Database Settings page

Click **Yes** on the Microsoft Certificate Services dialog box stating that Internet Information Services must be temporarily stopped.

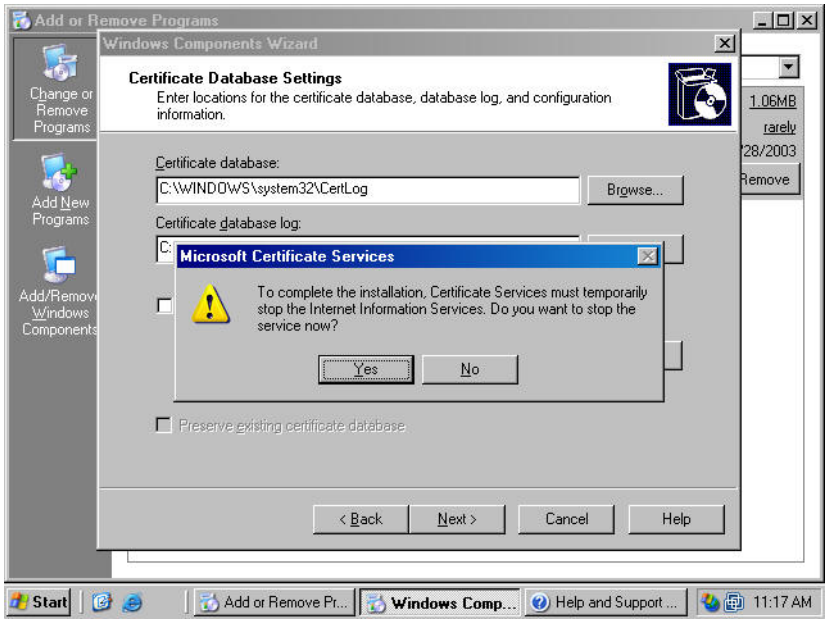

Figure 12 - Microsoft Certificate Services dialog box

Click **Yes** on the Microsoft Certificate Services dialog box stating that Active Server Pages must be enabled on IIS if Certificate Services Web enrollment site. Is to be used.

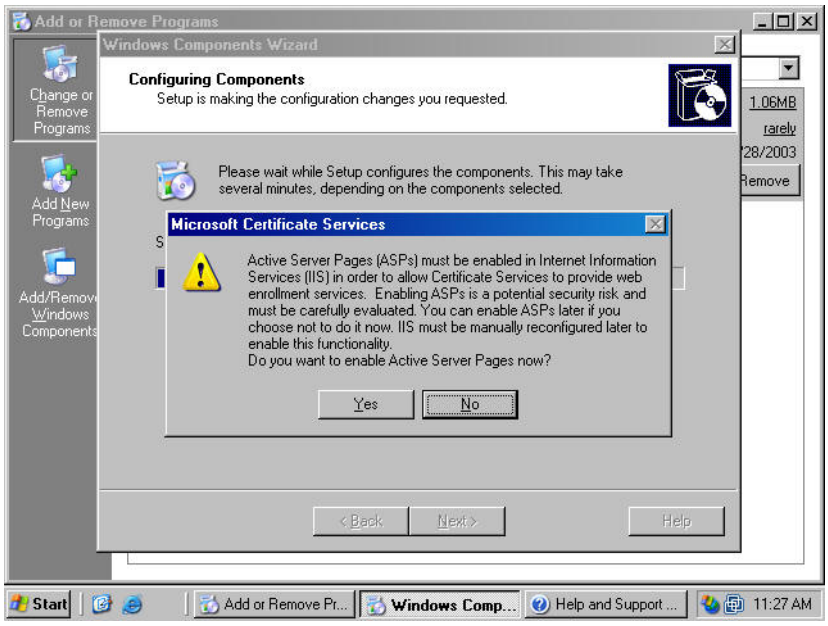

Figure 13 - Microsoft Certificate Services dialog box

Click **Finish** on the Completing the Windows Components Wizard page.

Figure 14 - Windows Components Wizard page

Close the Add or Remove Programs window.

Required Linux Modules

With the Windows side configured, install the appropriate libraries and modules on the Linux workstation. The following modules will be required for LDAP authentication and authorization:

- Openssl-0.9.7g-2.2

These modules can be installed by using the Software Management application located within the open SUSE Yast Control Center.

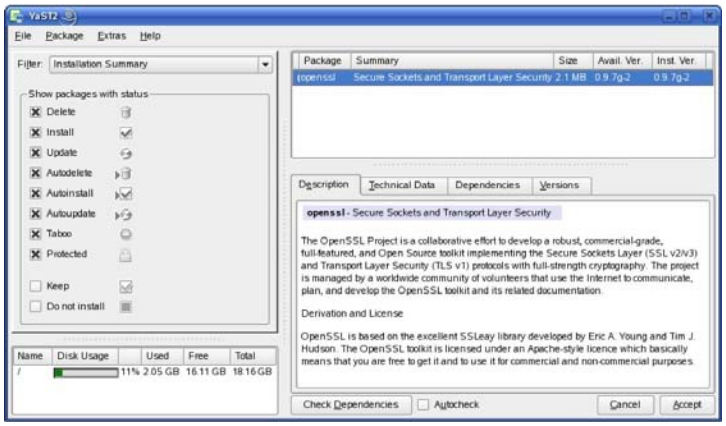

Figure 15 - SSL Module

Converting the CA certificate

Now that the lab is setup, the CA certificate will need to be acquired from the server. This can be done by downloading it from the certificate server web page located at http://*certificate _server*/certsrv/ where *certificate_server* can be either the hostname, FQDN or the IP address of the machine. Save it in the root directory on the Linux workstation.

Once the certificate is saved, it must now be converted to the PEM format. First,check that certificate is intact. This is done with the following command:

```
openssl x509 –inform DER –test < cacert.cer
```

An unencrypted dump of the certificate will be shown, to verify that the certificate is not corrupted. The output should look something like this:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            24:e0:16:13:57:33:65:ad:45:2a:b9:80:75:8d:c6:c0
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: DC=local, DC=devr2, CN=certtest
        Validity
            Not Before: Nov 29 18:33:07 2005 GMT
            Not After : Nov 29 18:42:48 2010 GMT
        Subject: DC=local, DC=devr2, CN=certtest
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (2048 bit)
                    .
                    .
                    .
```

Now that the certificate has been verified it should now be converted to the PEM format. This is done using the openssl command with the following syntax:

```
openssl x509 –inform DER –outform PEM –in cacert.cer –out cacert.pem
```

The output should be the new certificate in PEM format that the Linux workstation will use. To verify the ticket was converted successfully, dump the ticket to text once again with this command:

```
openssl x509 –inform PEM –test < cacert.pem
```

The output should look similar to the output before the conversion. Once output of the new certificate has been verified, it needs to be placed in the /etc/ssl/certs directory. Verify the file permissions are set for everyone to have read access on the certificate.

Edit /etc/ldap.conf

Now that the CA certificate is converted, /etc/ldap.conf needs to be edited. Amending the ldap.conf file that was used in the last section, with the addition three lines and changes to two other lines. The first line changed is the URI, change it from “ldap” to “ldaps”. The second line will be to turn the SSL feature on. The next line is to tell the PAM module where to find the CA certificate that was just converted. The adjacent line tells the client not to request the cert since we have already installed it on the local client. The last line is to change the port to 636 which is the default port for LDAPS. Here is the finished ldap.conf file.

```
uri                     ldaps://devdcr2aa.devr2.local
host                    devr2dcaa.devr2.local
base                    cn=Users,dc=devr2,dc=local
ldap_version            3
binddn                  cn=PADL,cn=Users,dc=devr2,dc=local
bindpw                  MicrosoftLinux
ssl                     yes
TLS_CACERT              /etc/ssl/certs/cacert.pem
TLS_REQCERT             never
port                    636
scope                   sub
timelimit               30
nss_map_objectclass     posixAccount User
nss_map_objectclass     shadowAccount User
nss_map_attribute       uid sAMAccountName
nss_map_attribute       uidNumber uidNumber
nss_map_attribute       gidNumber gidNumber
nss_map_attribute       cn sAMAccountName
nss_map_attribute       uniqueMember msSFU30PosixMemberOf
nss_map_attribute       userPassword unixUserPassword
nss_map_attribute       homeDirectory unixHomeDirectory
nss_map_attribute       loginShell loginShell
nss_map_attribute       gecos name
nss_map_objectclass     posixGroup Group
pam_login_attribute     sAMAccountName
pam_filter              objectclass=User
pam_password            ad
nss_base_passwd         cn=Users,dc=devr2,dc=local?sub
nss_base_shadow         cn=Users,dc=devr2,dc=local?sub
nss_base_group          cn=Users,dc=devr2,dc=local?sub
```

After modifying the file, save it and then test that it works using the “getent passwd” and “getent group” like we did in the previous section. If the tests were successful then reboot the Linux client and logon with and Active Directory account.

Samba using Winbind

What is Samba?

Samba is an Open Source/Free Software suite that provides seamless interoperability between Linux/Unix systems and Windows-based systems. With samba 3, support for Active Directory was added allowing Kerberos authentication and attribute mapping using LDAP queries; but unlike Windows NT 4.0 Domains, samba cannot be configured to an Active Directory Domain Controller at this time. Support for Microsoft's Distributed File System and integration of the Microsoft printing services to be integrated into a CUPS environment has been added. The biggest addition is the elimination of the need for shadow accounts on the Linux/Unix systems with the use of winbind

What is Winbind?

Winbind is a component of the Samba suite of programs that solves the unified logon problem. Winbind uses a UNIX implementation of Microsoft RPC calls, Pluggable Authentication Modules (PAM), and the name service switch (NSS) to allow Windows NT domain users to appear and operate as UNIX users on a UNIX machine.

Winbind provides three separate functions:

- Authentication of user credentials (via PAM).
- Identity resolution (via NSS).
- Winbind maintains a database called winbind_idmap.tdb in which it stores mappings between UNIX UIDs, GIDs, and NT SIDs.

What is Kerberos?

Kerberos is an authentication protocol which allows individuals communicating over an insecure network to prove their identity to one another in a secure manner. It was designed using a client-server model to provide mutual authentication — both the user and the service verify each other's identity to prevent eavesdropping or replay attacks, and ensure the integrity of the data. For more information on Kerberos can be found at the MIT website: http://web.mit.edu/kerberos/

Windows Configuration

Using Samba with Active Directory does not require any changes to the AD environment. Windows Server 2003 with SP1 can be used; Windows Server 2003 R2 is not needed on the servers for Samba to work. The installation and configuration of modules is done completely on the Linux side.

Required Linux Modules

The following modules will be required to be installed on the Linux system:

- Libsmbclient-3.0.20-4
- Samba-3.0.20-4
- Samba-client-3.0.20-4
- Samba-winbind-3.0.20-4
- Krb5-1.4.1-5
- Krb5-client-1.4.1-5
- Pam_krb5-2.2.0-6

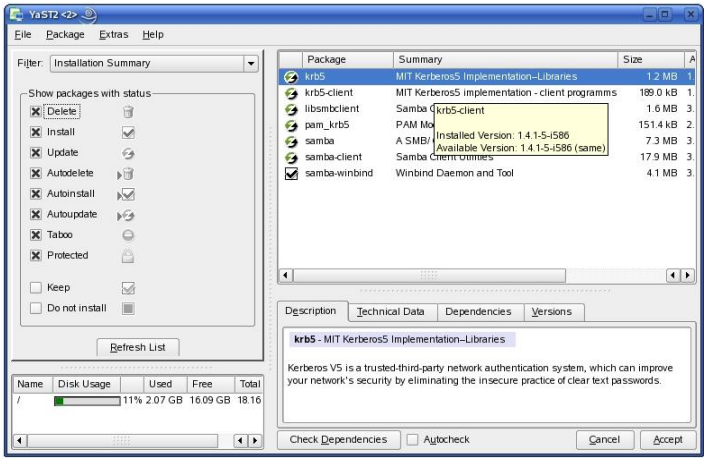

Figure 16 - Samba & Kerberos Modules

Configuring Kerberos

The Kerberos implemented by Microsoft in Windows is RFC 1510 compatible, but they have made several changes from the standard. The default encryption type used by Microsoft is the RC4-HMAC with a variable key-length. This was done for backwards compatibility and for export laws involving the use of DES encryption. The implementation of the Privilege Access Certificate (PAC) is a Microsoft proprietary field within the Kerberos ticket containing information about the client including the SID and group membership. Non-Microsoft clients should ignore this field except when the UDP packet becomes so large that TCP is required to send the data. Versions of Kerberos 5 before 1.3.1 were unable to resend the data using TCP and would fail.

So now that the necessary modules are installed, and with an understanding of Microsoft Kerberos; begin, by editing the /etc/krb5.conf file and adding the following information:

```
[libdefaults]
     default_realm = DEVR2.LOCAL
     clockskew = 300
     default_keytab_name = FILE:/etc/krb5.keytab
     dns_lookup_realm = false
     dns_lookup_kdc = true
     default_tgs_entypes = rc4-hmac des-cbc-md5
     default_tkt__enctypes = rc4-hmac des-cbc-md5
     permitted_enctypes = rc4-hmac des-cbc-md5
     dns_fallback = yes

[realms]
DEVR2.LOCAL = {
     kdc = devr2.local
     default_domain = devr2.local
     admin_server = devr2.local
```

```
        kpasswd_server = devr2.local
    }

    [logging]
        default = FILE:/var/log/krb5lib.log

    [domain_realm]
        .devr2.local = DEVR2.LOCAL
        devr2.local = DEVR2.LOCAL

    [appdefaults]
    pam = {
        ticket_lifetime = 1d
        renew_lifetime = 1d
        forwardable = true
        proxiable = true
        retain_after_close = true
        minimum_uid = 10000
        try_first_pass = true
    }
```

The first line [libdefaults] denotes the default library settings section of the configuration file. These settings will be applied to all realms unless noted differently within the realm section. The next line , denotes the default realm that will be used by a client to obtain a ticket. Moving on, we have the clockskew setting which denotes the amount of time variance that the client can have from the the KDC. The default value in Active Directory is 5 minutes, so set this to be the same - 300 seconds. The next line defines the name and location of the keytab file, by default it is /etc/krb5.keytab. The dns_lookup_realm indicates whether DNS TXT records should be used to determine the Kerberos realm of a host. The dns_lookup_kdc entry indicates whether DNS SRV records should be used to locate the KDCs and other servers for a realm, if they are not listed in the information for the realm. The next three lines define the encryption types that will be allowed, now remember from earlier in this section that Microsoft uses rc4-hmac encryption for backwards compatibility so use that and keep des-cbc-md5 for Linux compatibility as well. The last entry I this section is dns_fallback, which is a general flag controlling the use of DNS for Kerberos information. Some of these entries are redundant and actually reflect the default settings, but have been entered in here for general troubleshooting purposes.

The next section is the [realms] section, which contains subsections keyed by Kerberos realm names which describe where to find the Kerberos servers for a particular realm, and other realm-specific information. In this section there will be only one subsection for the DEVR2.LOCAL realm. Since every domain controller in an Active Directory environment is a KDC, the kdc entry can be set to the fully qualified domain name. Since it is possible for us to have a different realm name from our fully qualified domain name; the fully qualified domain name would be entered under the default_domain parameter. To make troubleshooting easier, keep the realm and the fully qualified domain name the same. The admin_server entry identifies the host where the administration server is running; since any domain controller could be used, just enter the fully qualified domain name. The same is true of the kpasswd_server field.

The [logging] section indicates how a particular entity is to perform its logging. The relations specified in this section assign one or more values to the entity name. Since this is a client the only entry we need is the default, which we tell it to log to a file located /var/log/krb5lib.log.

The [domain_realm] section contains relations which map sub-domains and domain names to Kerberos realm names. This is used by programs to determine what realm a host should be in, given its fully qualified domain name. Two entries are put in that state the same thing, there seems to be a lot of debate if both are needed; both are entered to eliminate it as a possible issue in testing.

The [appdefaults] section names a Kerberos V5 application or an option that is used by some Kerberos V5 applications. The application that we will be defining is PAM. In here defines some of the basic Kerberos ticket settings, for example ticket lifetime, renewal lifetime, can it be forwarded, etc.

Parameters highlighted in blue are dynamic; these will change to fit the environment the system in.

Configuring /etc/ntp.conf

Time plays a significant role with the use of Kerberos. The third line of our krb5.conf file deals with the difference of time between the KDC and the client. If it falls outside this skew the authentication attempt would fail. To prevent this use the Network Time Protocol (NTP) to receive the time from an authoritative source. In Active Directory every domain controller is a NTP server which allows clients to synchronize their time with the closest domain controller. Also, the domain controller holding the Primary Domain Controller (PDC) emulator flexible single master operations (FSMO) role is the authoritative time source for the domain by default.

In the lab there is just have one domain controller, enter it into the /etc/ntp.conf. This is done by finding and editing the following line:

```
server 172.16.126.183
```

Now just save the file and restart the service by typing:

```
linux:~ #/etc/rc.d/ntp restart
```

Configuring /etc/samba/smb.conf

The Samba configuration file holds all the configuration settings for the smbd service. This file is located in the /etc/samba directory and is named smb.conf. Like the krb5.conf file the smb.conf is also divided up into sections with specific entries for each section. The following is the smb.conf file that was used in the testing:

```
 [global]
    Workgroup = DEVR2
    realm = DEVR2.LOCAL
    security = ADS
    encrypt passwords = yes
    password server = devr2.local
    username map = /etc/samba/smbusers
    log level = 3
    log file = /var/log/samba/%m
    ldap ssl = no
    map to guest = Bad User
    idmap uid = 10000-20000
    idmap gid = 10000-20000
    template shell = /bin/bash
    template homedir = /home/%U
    use kerberos keytab = true
[homes]
    comment = Home Directories
    valid users = %S
    browseable = No
    read only = No
```

First, notice that there are two sections included in this file; global and homes. There are additional sections for defining and configuring the printing and other environments, but are outside the scope of this document.

The parameters in the [global] section apply to the system as a whole, or are the defaults for sections that do not specifically define certain items. The first four lines of this section are required in order to configure Samba 3 for Active Directory Domain membership using Kerberos authentication. The workgroup parameter is used to define the NetBIOS for the Active Directory domain. The next parameter, realm, is the name of the Kerberos realm; which is usually the fully qualified domain name of the Active Directory Domain. The security parameter is the most important option; it sets the security mode bit. This must be set to "ADS" for samba to use Kerberos authentication protocol. The encrypt passwords entry controls whether encrypted passwords will be negotiated with the client; and by default anything above Windows NT 4.0 SP3 will automatically encrypt the password.

The username map option gives the location of a file containing a mapping of usernames from the clients to the server. The map file is parsed line by line. Each line should contain a single UNIX username on the left then an equals (=), followed by a list of usernames on the right. This file will allow the mapping of certain accounts to Active Directory accounts.

Log level parameter sets the default debug level for samba if not specified when starting the service. This will help with troubleshooting, if omitted logging will not occur. The next line specifies the location of the debug log files.

The next entry is for "ldap ssl", which is set to no. The default setting if this option is omitted is to "start_tls". For testing purposes leave this option off, it can always setup SSL later on if needed.

The "map to guest" option can be used whenever the security mode is not set to share. This parameter can have three different values: never, bad user, bad password. User logins with an invalid password are rejected, unless the username does not exist, in which case it is treated as a guest login and mapped into the guest account; when set to "bad user". If set to "never" the user logon requests with an invalid password are rejected, this is the default option. The "bad password" option is not recommended due to the fact that it can mask configuration issues.

The idmap uid parameter specifies the range of user ids that are allocated for use in mapping UNIX users to NT user SIDs. This range of ids should have no existing local users within it as strange conflicts can occur. This parameter is synonymous with the winbind uid parameter.

The idmap gid parameter specifies the range of group ids that are allocated for use in mapping UNIX group to NT group SIDs. This range of ids should have no existing local groups within it as strange conflicts can occur. This parameter is synonymous with the winbind gid parameter.

Template shell parameter is used by the winbind daemon to fill in the login shell for that user. The equivalent is true for the template homedir parameter.

The final option in this section is the use kerberos keytab. This specifies whether Samba should attempt to maintain service principals in the systems keytab file. The default setting is false.

The next section, named [homes], when included in the configuration file, services connecting clients to their home directories can be created on the fly by the system.

The "valid users" option defines a list of users that should be allowed to login to the service. If left blank, which is the default, then any user can login. The current service name is substituted for the %S.

The next parameter, "browseable", controls whether this share is seen in the list of available shares in a net view and in the browse list.

The last option, “read only”, defines if users can create or modify files in the service’s directory. By default, this is enabled.

To validate and check wer smb.conf file for errors, use the “testparm” command. Use the following syntax:

```
linux:~ # testparm –s
```

If any errors were found correct them and run the command again, otherwise proceed on.

Creating a Computer Account

With Kerberos and Samba configured, join Linux system to the Active Directory domain. The first step in this process is to start a console window as the “root” user, this process must be run by the root account (UID=0).

The next step will be to join the computer to Active Directory, this is accomplished by using the “net ads join -U” command. This command must be used in order to use Kerberos and LDAP; if the similar command of “net rpc join” is used instead, NTLM authentication will be used and the system will act similar to a Windows NT 4.0 Server. Here is the command syntax:

```
linux:~ # net ads join -Uadministrator
administrator's password:
Using short domain name -- DEVR2
Joined 'LINUX' to realm 'DEVR2.LOCAL'
linux:~ #
```

Configuring /etc/nsswitch.conf

In order to map the Unix attributes to Active Directory edit the following three lines in the /etc/nsswitch.conf:

```
passwd:     files winbind
group:      files winbind
hosts:      files dns
```

Once completed, test this configuration by issuing the “getent passwd” and the “getent group” commands. If successful a list of Linux user accounts and groups, but Active Directory accounts and groups as well, as shown below.

```
linux:/etc # getent passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
            .
            .
DEVR2\administrator:x:10000:10000:Administrator:/home/U%:/bin/bash
DEVR2\guest:x:10001:10001:Guest:/home/U%:/bin/bash
DEVR2\krbtgt:x:10003:10000:krbtgt:/home/U%:/bin/bash
DEVR2\dsmith:x:10005:10000:Doug Smith:/home/U%:/bin/bash
DEVR2\padl:x:10009:10000:PADL:/home/U%:/bin/bash
DEVR2\linux$:x:10011:10002:linux2:/home/U%:/bin/bash
linux:/etc # getent group
root:x:0:
bin:x:1:daemon
daemon:x:2:
            .
            .
nogroup:x:65534:nobody
users:x:100:
DEVR2\domain computers:x:10002:
DEVR2\domain controllers:x:10003:
DEVR2\cert publishers:x:10006:
DEVR2\domain admins:x:10007:
```

```
DEVR2\domain users:x:10000:
linux:/etc #
```

Configuring PAM

To configure the authentication piece, modify the PAM configuration file. For this test, modification of the XDM PAM module was done, so that if a mistake were to occur access via command line is still available. Below is a listing of the /etc/pam.d/xdm file:

```
#%PAM-1.0
auth        required    pam_env.so
auth        sufficient  pam_winbind.so
auth        required    pam_unix2.so nullok use_first_pass
account     requisite   pam_time.so
account     sufficient  pam_winbind.so
account     required    pam_unix2.so
session     required    pam_mkhomedir.so skel=/etc/skel umask=0027
password    required    pam_pwcheck.so nullok
password    optional    pam_winbind.so use_first_pass use_authtok
password    required    pam_unix2.so nullok use_first_pass
use_authtok
```

With this completed, reboot the system and logon with an Active Directory account.

Kerberos with LDAP

Method Summary

In this section the use of Kerberos with LDAP methods will be combined for authentication and authorization. Kerberos protocol will be used to do an initial authentication with Active Directory; then the use of the LDAP protocol to bind to Active Directory for our mappings.

Required Linux Modules

The following Linux modules will need to be installed for Kerberos and LDAP authentication and authorization:

- openldap-client-2.2.27-6
- ldapcipplib-0.0.3-33
- nss_ldap-238-2
- pam_ldap-178-3
- libnscd-1.1-5
- nscd-2.3.5-40
- Openssl-0.9.7g-2.2
- Krb5-1.4.1-5
- Krb5-client-1.4.1-5
- Pam_krb5-2.2.0-6

These modules can either be downloaded and compiled from their source code or installed from within the open SUSE Yast Control Center.

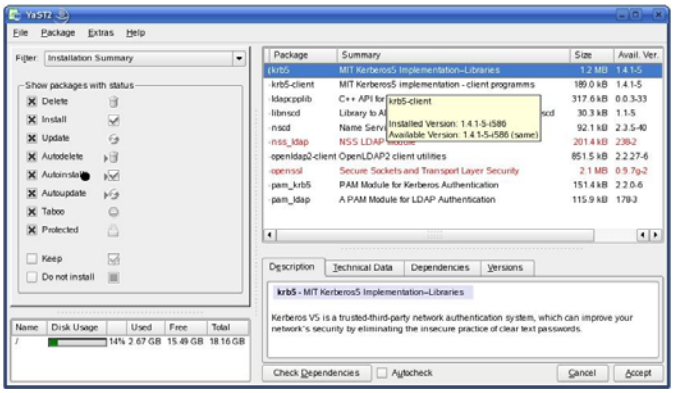

Figure 17 - Kerberos/LDAP/SSL Modules

Configuring /etc/krb5.conf

The setup and configuration of Kerberos is identical to that used in the last section, Samba using Kerberos and LDAP. Below is a copy of the krb5.conf file.

```
[libdefaults]
     default_realm = DEVR2.LOCAL
     clockskew = 300
     default_keytab_name = FILE:/etc/krb5.keytab
     dns_lookup_realm = false
     dns_lookup_kdc = true
     default_tgs_entypes = rc4-hmac des-cbc-md5
     default_tkt__enctypes = rc4-hmac des-cbc-md5
     permitted_enctypes = rc4-hmac des-cbc-md5
     dns_fallback = yes

[realms]
DEVR2.LOCAL = {
     kdc = devr2.local
     default_domain = devr2.local
     admin_server = devr2.local
     kpasswd_server = devr2.local
}

[logging]
     default = FILE:/var/log/krb5lib.log

[domain_realm]
     .devr2.local = DEVR2.LOCAL
     Devr2.local = DEVNR2YS.LOCAL

[appdefaults]
pam = {
     ticket_lifetime = 1d
     renew_lifetime = 1d
     forwardable = true
     proxiable = true
     retain_after_close = true
     minimum_uid = 10000
     try_first_pass = true
}
```

Configuring /etc/ntp.conf

In the lab there is just have one domain controller, enter it into the /etc/ntp.conf. This is done by finding and editing the following line:

```
server 172.16.126.183
```

Creating a Keytab

For Kerberos authentication we will need to add a user account in Active Directory for the Linux workstation. After we create the account we will need to edit some attributes under the account object. From the “Account” tab, under the account options we will need to check the following boxes:

- Password never expires
- Use DES encryption types for this account
- Do not require Kerberos preauthentication

Once this is done, we will need to change the password for the user account for the DES encryption to take effect.

To create the keytab file will require we to use a program named “ktpass”, which is located in the Windows Server 2003 Support Tools. To create my keytab files for my machine named linux, we used the following syntax:

```
C:\>ktpass /out c:\linux.keytab /princ
host/linux.devr2.local@DEVR2.LOCAL /pass * /mapuser linux /crypto
DES-CBC-CRC
```

Now let’s explain the syntax in the preceding line. The “out” switch specifies where the key tab file will be saved along with its name. Our service principal is listed after the “princ” switch and should be in the form of *service/fqdn@REALM*. Next comes out password using the “pass” switch. Using the “*” which will have the program prompt me for the password. Using the “mapuser” switch tells the program to which account we want to add this service principal to. Finally, with the “crypto” switch we designate the encryption type for the keytab file; DES-CBC-CRC is the default for MIT Kerberos.

We should then get the following output if successful:

```
Targeting domain controller: devdcr2aa.devr2.local
Successfully mapped host/linux.devr2.local@DEVR2.LOCAL to linux
WARNING: pType and account type do not match.  This might cause
problems.
Key created.
Output keytab to c:\linux.keytab:
Keytab version: 0x502
keysize 69 host/linux.devr2.local@DEVR2.LOCAL ptype 0
(KRB5_NT_UNKNOWN)
vno 4 etype 0x1 (DES-CBC-CRC) keylength 8 (0x579785833dfd1907)
```

With the keytab file created, we will need to copy it to the Linux workstation. Once there, we will need to import the keytab file into Linux using the ktutil program to import and save the file. From a terminal window as root, we will need to run the ktutil program. Next, we will need to read the keytab file that we created, this is the import. Now we will just give it a quick visual check by listing it. If all looks good we will then save the information by writing it to /etc/krb5.keytab. An example of this is shown below:

```
linux~ # ktutil
ktutil: rkt <host>.keytab
ktutil: list
slot KVNO Principal
---- ---- ---------------------------------------
   1    4 host/linux.devr2.local@DEVR2.LOCAL
ktutil: wkt /etc/krb5.keytab
ktutil: quit
```

With that complete we will need to test our configuration to make sure it is correct and the system is able to authenticate against the KDC. This is done by simply typing the command “kinit –k” at a terminal window. If successful, we should not get any returned messages. We may look at the principal by typing the command “klist –k”.

```
linux~ # kinit –k
linux~ # klist –k
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
---- ---------------------------------------
   4 host/linux.devr2.local@DEVR2.LOCAL
```

Configuring /etc/crontab

With Kerberos configured and tested, we can now move on to setting up a local service ticket cache for the LDAP queries. This is done by adding the following line to the /etc/crontab file:

```
**/2*** /usr/bin/kinit -k -c /etc/.ldapcache –S \
ldap/devdcr2aa.devr2.local@DEVR2.LOCAL \
host/linux.devr2.local@DEVR2.LOCAL && chmod a+r /etc/.ldapcache
```

Every two hours, the system will get a service ticket for ldap on the server devdcr2aa, and place it in a cache file named ".ldapcache" located in the /etc directory with everyone having read permissions to the file.

Configuring /etc/ldap.conf

In the first section we used LDAP for both authentication and authorization. We will use the same configuration. We will leave the entries for using SSL encryption commented out for now; they can be enabled later for added security.

```
uri                         ldap://devdcr2aa.devr2.local
host                        devr2dcaaaa.devr2.local
base                        cn=Users,dc=devr2,dc=local
ldap_version                3
binddn                      cn=PADL,cn=Users,dc=devr2,dc=local
bindpw                      MicrosoftLinux
ssl                         off
#TLS_CACERT                 /etc/ssl/certs/cacert.pem
#TLS_REQCERT                never
port                        389
scope                       sub
timelimit                   30
nss_map_objectclass         posixAccount User
nss_map_objectclass         shadowAccount User
nss_map_attribute           uid sAMAccountName
nss_map_attribute           uidNumber uidNumber
nss_map_attribute           gidNumber gidNumber
nss_map_attribute           cn sAMAccountName
nss_map_attribute           uniqueMember msSFU30PosixMemberOf
nss_map_attribute           userPassword unixUserPassword
nss_map_attribute           homeDirectory unixHomeDirectory
nss_map_attribute           loginShell loginShell
nss_map_attribute           gecos name
nss_map_objectclass         posixGroup Group
pam_login_attribute         sAMAccountName
pam_filter                  objectclass=User
pam_password                ad
nss_base_passwd             cn=Users,dc=devr2,dc=local?sub
nss_base_shadow             cn=Users,dc=devr2,dc=local?sub
nss_base_group              cn=Users,dc=devr2,dc=local?sub
```

Configuring /etc/nsswitch.conf

In order for us to map the Unix attributes to Active Directory we just need to edit the following three lines in the /etc/nsswitch.conf:

```
passwd:     files ldap
group:      files ldap
hosts:      files dns
```

Once completed, we can test this configuration by issuing the "getent passwd" and the "getent group" commands. If successful we should not only see a list of Linux user accounts and groups, but Active Directory accounts and groups as well.

Configuring PAM

To configure the authentication piece, we will need to modify the PAM configuration. For my test, I chose to modify the XDM PAM module so that if a mistake were to occur I could still get in via command line access. Below is a listing of my /etc/pam.d/xdm file:

```
#%PAM-1.0
auth        required    pam_env.so
auth        sufficient  pam_krb5.so
auth        required    pam_unix2.so nullok use_first_pass
account     requisite   pam_time.so
account     sufficient  pam_krb5.so
account     required    pam_unix2.so
session     required    pam_mkhomedir.so skel=/etc/skel umask=0027
password    required    pam_pwcheck.so nullok
password    optional    pam_krb5.so use_first_pass use_authtok
password    required    pam_unix2.so nullok use_first_pass
```

With this completed, we will now reboot the system and should be able to logon with an Active Directory account.

Additional Security

Earlier in the configuration, we had commented out several lines in the /etc/ldap/conf. These lines were for enabling the use SSL with LDAP. While the username and password are encrypted via Kerberos, there is still other information that could be acquired via a network sniff. Most importantly is the LDAP binding account and password, this information is still transmitted in clear text. To use SSL encryption we must get and convert the CA certificate, which is outlined in the second section. Once that is complete we will need to make some changes to wer /etc/ldap.conf. The ldap.conf is listed below with the changes in bold test.

```
uri                     ldaps://devdcr2aa.devr2.local
host                    devr2dcaaaa.devr2.local
base                    cn=Users,dc=devr2,dc=local
ldap_version            3
binddn                  cn=PADL,cn=Users,dc=devr2,dc=local
bindpw                  MicrosoftLinux
ssl                     on
TLS_CACERT              /etc/ssl/certs/cacert.pem
TLS_REQCERT             never
port                    636
scope                   sub
timelimit               30
nss_map_objectclass     posixAccount User
nss_map_objectclass     shadowAccount User
nss_map_attribute       uid sAMAccountName
nss_map_attribute       uidNumber uidNumber
nss_map_attribute       gidNumber gidNumber
nss_map_attribute       cn sAMAccountName
nss_map_attribute       uniqueMember msSFU30PosixMemberOf
nss_map_attribute       userPassword unixUserPassword
nss_map_attribute       homeDirectory unixHomeDirectory
nss_map_attribute       loginShell loginShell
nss_map_attribute       gecos name
nss_map_objectclass     posixGroup Group
pam_login_attribute     sAMAccountName
pam_filter              objectclass=User
pam_password            ad
nss_base_passwd         cn=Users,dc=devr2,dc=local?sub
nss_base_shadow         cn=Users,dc=devr2,dc=local?sub
nss_base_group          cn=Users,dc=devr2,dc=local?sub
```

Once the file is changed and saved, we will need to test to verify that it is functioning properly.

Summary of Methods

	Advantage	Disadvantage
LDAP	-Uses Standard Protocols	-All information in plaintext -Requires a bind account or anonymous access to Active Directory -Bind account and password sent in plaintext -Require Windows Server 2003 R2 Schema change
LDAPS	-Uses Standard Protocols -All information encrypted using SSL	-Requires a bind account or anonymous access to Active Directory -Requires an Enterprise Certificate Authority to be setup. -Require Windows Server 2003 R2 Schema change
SAMBA using Winbind	-Uses Standard Protocols -Does not require Windows Server 2003 R2 Schema change	-If not configured correctly, it will use NTLM authentication over RPC.
Kerberos and LDAP	-Uses Standard Protocols -Allows for impersonation	-Requires a bind account or anonymous access to Active Directory -Bind account and password sent in plaintext -Require Windows Server 2003 R2 Schema change
Kerberos and LDAPS	-Uses Standard Protocols -Allows for impersonation	-Requires a bind account or anonymous access to Active Directory -Requires an Enterprise Certificate Authority to be setup. -Require Windows Server 2003 R2 Schema change